Implementation of an Internet-based remote controller with guaranteed exponential stabilization

Wenjuan Jiang, Jean-Pierre Richard, Armand Toguyeni

IMPLEMENTATION OF AN INTERNET-BASED REMOTE CONTROLLED ROBOT WITH GUARANTEED EXPONENTIAL STABILIZATION

Wenjuan Jiang * **Jean-Pierre Richard** * **Armand Toguyeni** *

* *Wenjuan Jiang, Jean-Pierre Richard and Armand Toguyeni are with LAGIS CNRS UMR 8146, Ecole Centrale de Lille, BP 48, 59651 Villeneuve d'Ascq Cedex, France. J.P. Richard is also with the project ALIEN, INRIA Futurs. wenjuan.jiang@ec-lille.fr, armand.toguyeni@ec-lille.fr and jean-pierre.richard@ec-lille.fr*

Abstract: In this article, an Internet-based remote control system is designed and implemented. The communication is based on the Master-Slave structure. The Master PC communicates with the Slave from about 40km away by UDP protocol. In order to guarantee the Master and Slave clocks to be synchronized, the NTP (Network Time Protocol) is used in both sides. The packets are sent together with time-stamps. The controller design (master) relies on a remote observer that achieves a state prediction of the application (slave), despite the variable communication delays. The Slave comprises a PC and a robot Miabot of Merlin company. The protocol Bluetooth is used between the local PC and the robot. Internet-based remote systems are subject to variable time delays (including communication and data-sampling delays) and data packets losses (due to the unstable Internet network). We have continuously tested the RTT (round-trip-time) between the two PCs in the daytime and nighttime by the protocol ICMP (Internet Control Message). From these tests, an evaluation of the maximal time delay is obtained. Our structure allows one to guarantee an exponential stabilization performance, which is proven via a Lyapunov-Krassovski functional technique and involves the estimated delay upperbound. This means that the guaranteed decay rate is computed (via some LMI optimization) in relation to some maximal value of the communication delays. Of course, for greater delay values, the performance cannot be guaranteed anymore and an alternative solution has to be considered. In our system, we give a command for the robot to stop until the communication comes back to a sufficient quality.

Keywords: Networked control, remote control, remote observer, Internet, UDP, time delay, exponential stability.

1. INTRODUCTION

As Internet is well developed, remote control system has been widely used in industrial, communicational, medical and even biological systems. However, alongside the advantage of low costs, the Internet inevitably brings problems to the closed-loop controlled system, such as delay variation, data-packets loss (M. Yu *et al.*, 2004) and disorder, which cause poor performance (J.P. Richard and T. Divoux, 2007), instability or danger. How to diminish the effect of time delay in the remote system is critical in the system design. The main solution can split into two (combinable) strategies (J.P. Richard and T. Divoux, 2007; J. Chiasson

[1] The work is partially supported by the CSC (China Scholarship Council) and Region Nord Pas-de-Calais (ARCIR RoboCoop).

and J.J. Loiseau, 2007): 1) Increase the network performances(QoS) or 2) design an adapted control that can compensate the network influence. The former focuses on the control of the network (protocol, route choosing, congestion strategy, etc.). The second approach is related to robust control and depends on the kind of network to be used (Internet, Ethernet...). In this article, we consider this last approach for an Internet network. Our aim is to ensure suitable stabilization and speed performances, i.e. exponential stabilization, despite the dynamic variations of the network. Note that, in the Internet case, the network delays cannot be modeled nor predicted. Moreover, the (variable) transmission delays are asymmetric, which means that the delay $h_1(t)$ from Master to Slave (shortly, M-to-S), and the return one (S-to-M) $h_2(t)$ normally satisfy $h_1(t) \neq h_2(t)$. Because of this lack of knowledge, predictor-based control laws (E. Witrant *et al.*, 2007) cannot be applied. A delay maximizing strategy (J.P. Richard, 2003; A. Lelevé *et al.*, 2001) ("virtual delay", "buffer", or "waiting" strategy) can be carried out so to make the delay constant and known. This requires the knowledge of the maximum delay values $h_m \geq h_1(t)$ and $h_m \geq h_2(t)$. However, it is obvious that maximizing the delay up to its largest value decreases the speed performance of the remote system.

Our solution relies on the theoretical results of (A. Seuret *et al.*, 2004) (exponential stabilization of systems with unknown, varying delays), as well as (A. Seuret *et al.*, 2006) (networked control), the main lines of which will be shortly recalled in the next section. It allows for applying a waiting strategy only to the M-to-S communication, whereas the S-to-M communication takes the information into account as soon as received. The additional contribution of this paper is the design of an adapted computer implementation (M/S structure) based on the UDP protocol and involving lists as buffers. The choice of UDP is preferred to TCP because in our NCS (Networked Control System)situation, packets re-emitting is not needed and setting up the TCP connection between two PCs is time-consuming.

2. FEATURES OF THE REMOTE SYSTEM

The remote system is based on Master-Slave structure. In order to simplify the work of the Slave, the control and observation complexity is concentrated on the Master. The main features of the system refer to Fig.1. In the system, the robot Miabot of the company Merlin

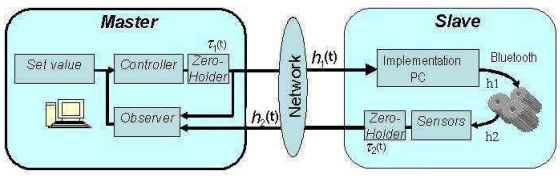

Fig. 1. Structure of the global system

Systems Corp. Ltd together with a PC serve as the Slave. The Miabot works as an accessory device which communicates with the PC by the port of Bluetooth. As showed in the Appendix Fig.A.1, the local delay of Bluetooth can be regarded as a constant one. In order to simplify our problem, we take the maximum local delay and add $h1$ and $h2$ into the respectively variable delay $h_1(t)$ and $h_2(t)$.

2.1 The three delay sources

In such a situation, the variable delays come from: 1) the communication through the Internet; 2) the data-sampling and 3) the possible packet losses. In the sequel, $h_1(t)$ and $h_2(t)$ denote the communication delays and $\tau_1(t)$ and $\tau_2(t)$ include the sampling delays and possible packet losses. The total Master-to-Slave delay $\delta_1(t)$ results from the addition of $h_1(t_k)$ and $\tau_1(t)$.

1) Both computers dates are synchronized before the system works. The NTP(Network Time Protocol)(D.L. Mills, 1995) is used in both sides according to the same pool time server. By this way, whenever the Master or the Slave receives the data including the time stamp, it knows the instant t_k of data sending out and the resulting transmission delay $h_i(t_k)$.

2) The real remote system, including Master, Slave and network, must involve some data sampling. However, following (A. Seuret *et al.*, 2005; E. Fridman *et al.*, 2004), this phenomenon is equivalent to a time-varying, discontinuous delay $\tau_i(t)$ (to be defined in (1)), which allows for keeping a continuous-time model. If the packets exchange between the Master and the Slave is of high speed, then $\tau_i(t)$ constitutes a disturbance that should be considered in the stabilization design (M. Yu *et al.*, 2004). $\tau_i(t)$ is variable but it is supposed there is a known T (maximum sampling period) so that $\tau_i(t) \leq T$.

3) If some packet p_{t_k} containing the sample at t_k is lost, or arrives later than the packet $p_{t_{k+1}}$, then the Master only considers the most recent data (i.e., those from $p_{t_{k+1}}$). If it is assumed that the maximum number of successive packets that can be lost is N, then the maximum resulting delay is nT. The same lines also holds for the control packets. From 2) and 3), the delay $\delta_i(t)$ has a known maximum $\delta_i^m(t) = (N+1)T + h_m$ and the delay variation satisfies $\dot{\delta}_i(t) \leq 1$. In order to keep simple expressions, the notation T will be kept preferably to $T' = (N+1)T$.

Summarizing, given a signal $g(t)$ and the global delay $\delta(t)$ which represents the combination of the sampling and packet loss delay with the delay $h(t_k)$ that the transmission line subjects to the packet containing the k^{th} sample at time t_k, $g(t)$ can be written as:

$$\begin{aligned} g(t_k - h(t_k)) &= g(t - h(t_k) - (t - t_k)), \\ &= g(t - \delta(t)), \\ t_k \leq t < t_{k+1},\ \delta(t) &\triangleq h(t_k) + t - t_k. \end{aligned} \tag{1}$$

4) Of course, for greater delay values, the performance cannot be guaranteed anymore and an alternative solution has to be considered. In our system, we give a command for the robot to stop until the communication comes back to a sufficient quality.

2.2 The control law

The controller computes a control law which takes into account some set value $c(t)$ to be reached by the Slave's state $x(t)$. Since $x(t)$ is not available to the Master, the state feedback control $u(t)$ is defined from a state estimate $\hat{x}(t)$ computed by an observer:

$$u(t) = K(\hat{x}(t) - c(t)). \tag{2}$$

A main difficulty is to determine the linear gain K of the state feedback control so to guaranty the exponential stability of the Slave motion despite the time-varying delay $\delta_1(t)$. Note this delay is not known by the Master when its control data is sent. In the next section, this problem is solved by using the LMI-based results of (A. Seuret *et al.*, 2004; A. Seuret *et al.*, 2006).

2.3 Transmission and receipt of the control data

The k^{th} data sent by the Master to Slave includes the control $u(t_{1,k})$ together with the time stamp when the packet is sent out. At time $t^r_{1,k}$, when the Slave receives the data it can calculate the time delay because of the time stamp. If the delay is greater than h_{1m}, the Slave should apply immediately the command.

The control u, sent out by the Master at time $t_{1,k}$, is received by the Slave at time $t^r_{1,k} > t_{1,k}$. It will be injected in the Slave input only at the pre-defined "target time" $t^{target}_{1,k} = t_{1,k} + h_{1m}$. The corresponding waiting time h_{1m} is depicted on Fig. 2. This is realistic because the transmission delay is bounded by a known value h_{1m}. By this way, the Master knows the time $t_{1,k} + h_{1m}$ when this control $u(t_{1,k})$ will be injected at the Slave input.

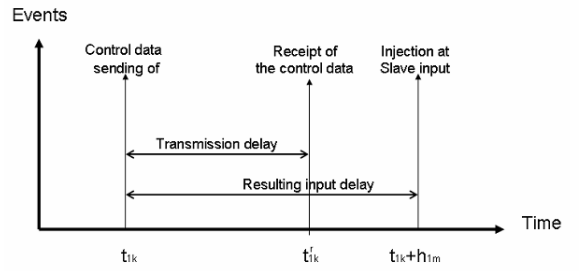

Fig. 2. Control data processing

3. CONTROL AND ESTIMATION DESIGN

3.1 Control objective

The main purpose of the system is to guaranty the robustness and speed performance of the global M/S system. In order to guaranty the closed-loop stability and speed rate whatever the delay variation, the exponential stability with the rate α must be achieved. In other words, there must be a real $\kappa \geq 1$ so that the solution $x(t;t_0,\phi)$ starting at any time t_0 from any initial function ϕ satisfies: $\|x(t;t_0,\phi)\| \leq \kappa \|\phi\| e^{-\alpha(t-t_0)}$.

For simplicity, the Slave is considered as a linear system. It is described as following form, in which (A,B,C) is observable.

$$\begin{cases} \dot{x}(t) = Ax(t) + Bu(t - \delta_1(t)), \\ y(t) = Cx(t), \end{cases} \tag{3}$$

The α-stabilization results we use for the controller and observer gains design refer to (A. Seuret *et al.*, 2004; A. Seuret *et al.*, 2005).

3.2 Observer design

For a given k and for any $t \in [t_{1,k} + h_{1m}\ ,\ t_{1,k+1} + h_{1m}[$, there exists a k' such that the proposed observer is of the form:

$$\begin{cases} \dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t_{1,k}) - L(y(t_{2,k'}) - \hat{y}(t_{2,k'})), \\ \hat{y}(t) = C\hat{x}(t). \end{cases} \tag{4}$$

The index k' corresponds to the most recent output information the Master has received. Note that the Master knows the time $t_{1,k}$ and the control $u(t_{1,k})$ (see Section 2.3), which makes this observer realizable.

Using the delay re-writing (1), one obtains:

$$\begin{cases} \dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t - \delta_1(t)) \\ \qquad\qquad -L(y(t - \delta_2(t)) - \hat{y}(t - \delta_2(t))), \\ \hat{y}(t) = C\hat{x}(t), \end{cases} \tag{5}$$

with $\delta_1(t) \triangleq h_{1,k} + t - t_{1,k}$ and $\delta_2(t) \triangleq h_{2,k'} + t - t_{2,k'}$. In other words, the observer is realizable because the times $t_{1,k}$ and $t_{2,k'}$ defining the observer delays are known thanks to the time stamps. The system features lead to $\delta_1(t) \leq h_{1m} + T$ and $\delta_2(t) \leq h_{1m} + T$.

We define the error vector between the estimated state $\hat{x}(t)$ and the present system state $x(t)$ as $e(t) = x(t) - \hat{x}(t)$. From (3) and (5), this error is ruled by:

$$\dot{e}(t) = Ae(t) - LCe(t - \delta_2(t)). \tag{6}$$

Theorem 1. (A. Seuret *et al.*, 2006) Suppose that, for some positive scalars α and ε, there exists $n \times n$ matrices $0 < P_1$, P, S, Y_1, Y_2, Z_1, Z_2, Z_3, R, R_a and a matrix W with appropriate dimensions such that the following LMI conditions are satisfied for $j = 1, 2$:

$$\begin{bmatrix} \Psi_2 & \begin{bmatrix} \beta_{2j}WC - Y_1 \\ \varepsilon\beta_{2j}WC - Y_2 \end{bmatrix} & \mu_2\beta_{2j}\begin{bmatrix} WC \\ \varepsilon WC \end{bmatrix} \\ * & -S & 0 \\ * & * & -\mu_2 R_a \end{bmatrix} < 0,$$

$$\begin{bmatrix} R & Y \\ * & Z \end{bmatrix} \geq 0,$$

where β_{2j} are defined by:

$$\begin{array}{ll} \beta_{11} = e^{\alpha(\delta_1-\mu_1)}, & \beta_{12} = e^{\alpha(\delta_1+\mu_1)}, \\ \beta_{21} = e^{\alpha(\delta_2-\mu_2)}, & \beta_{22} = e^{\alpha(\delta_2+\mu_2)}, \end{array} \quad (7)$$

and the matrices Y, Z and Ψ_2 are given by:

$$Y = [Y_1\ Y_2], \quad Z = \begin{bmatrix} Z_1 & Z_2 \\ * & Z_3 \end{bmatrix}, \quad (8)$$

$$\begin{aligned} \Psi_2^{11} &= P^T(A_0+\alpha I) + (A_0+\alpha I)^T P + S \\ &\quad + \delta_2 Z_1 + Y_1 + Y_1^T, \\ \Psi_2^{12} &= P_1 - P + \varepsilon P^T (A_0+\alpha I)^T + \delta_2 \bar{Z}_2 + \bar{Y}_2, \\ \Psi_2^{22} &= -\varepsilon(P+P^T) + \delta_2 \bar{Z}_3 + 2\mu_2 R_a. \end{aligned}$$

Then, the gain:

$$L = (P^T)^{-1} W, \quad (9)$$

makes the error (6) of observer (5) exponentially converge to the solution $e(t) = 0$, with a decay rate α.

3.3 Control design

We first consider a controller $u = Kx$, *i.e.* the ideal situation $e(t) = 0$, $x(t) = \hat{x}(t)$ and:

$$\dot{x}(t) = Ax(t) + BKx(t - \delta_1(t)). \quad (10)$$

Theorem 2. (A. Seuret *et al.*, 2006) Suppose that, for some positive numbers α and ε, there exists a positive definite matrix $\bar{P}_1$, matrices of size $n \times n$: $\bar{P}$, $\bar{U}$, $\bar{Z}_1$, $\bar{Z}_2$, $\bar{Z}_3$, $\bar{Y}_1$, $\bar{Y}_2$ similarly to (8) and a $n \times m$ matrix W, such that the following LMI conditions hold:

$$\Gamma_{3i} = \begin{bmatrix} \Psi_3 & \begin{bmatrix} \beta_i BW - \bar{Y}_1^T \\ \varepsilon\beta_{1i} BW - \bar{Y}_2^T \end{bmatrix} & \mu_1 \begin{bmatrix} \beta_{1i} BW \\ \varepsilon\beta_{1i} BW \end{bmatrix} \\ * & -\bar{S} & 0 \\ * & * & -\mu_1 \bar{R}_a \end{bmatrix} < 0,$$

$$\forall i = 1,2,$$

$$\begin{bmatrix} \bar{R} & \bar{Y}_1 & \bar{Y}_2 \\ * & \bar{Z}_1 & \bar{Z}_2 \\ * & * & \bar{Z}_3 \end{bmatrix} \geq 0,$$

where β_{1i}, for $i = 1,2$, are defined by (7) and

$$\begin{aligned} \bar{\Psi}_3^{11} &= (A_0+\alpha I)\bar{P} + \bar{P}^T (A_0+\alpha I)^T + \bar{S} \\ &\quad + \delta_1 \bar{Z}_1 + \bar{Y}_1 + \bar{Y}_1^T, \\ \bar{\Psi}_3^{12} &= \bar{P}_1 - \bar{P} + \varepsilon \bar{P}^T (A_0+\alpha I)^T + \delta_1 \bar{Z}_2 + \bar{Y}_2, \\ \bar{\Psi}_3^{22} &= -\varepsilon(\bar{P}+\bar{P}^T) + \delta_1 \bar{Z}_3 + 2\mu_1 \bar{R}_a. \end{aligned}$$

Then, the gain:

$$K = W \bar{P}^{-1},$$

exponentially stabilizes the system (10) with the decay rate α for all delay $\delta_1(t)$.

3.4 Global stability of the remote system

The gains K and L have to be computed in such a way they exponentially stabilize the global Master-Slave-Observer system despite the variable delays $\delta_1(t)$ and $\delta_2(t)$. This global system is:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + BKx(t - \delta_1(t)) + BKe(t - \delta_1(t)), \\ \dot{e}(t) &= Ae(t) + LCe(t - \delta_2(t)). \end{aligned}$$

A separation principle is then applied to the previous system. Then if L and K are chosen with respect to sections 3.2 and 3.3 respectively, the global system is exponentially stable with the lowest decay rate.

4. IMPLEMENTATION OF THE REMOTE CONTROL SYSTEM

The transmission protocol UDP is applied to communicate the data between Master and Slave. In order to know the instant of data-sent, time stamps are added to the data packets. The data structure of list served as buffers is introduced for the program to search for the data of the right instance. In all the lists, the control data are restored in the decreasing order of its sending time. That is to say, the most recent sent data is always at the head of the list.

4.1 The structure of the Master

In order to implement the model for the remote control system, four-thread program is designed to fulfill the functions of Controller and Observer of Fig.1.

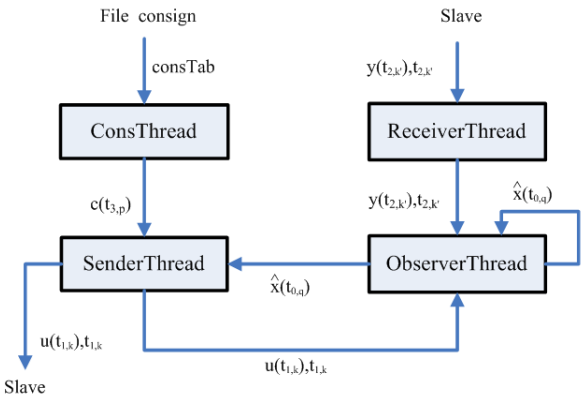

Fig. 3. Structure of the Master

These four threads are parallel working as showed in Fig.3. There are two buffers, *list_U* and *list_X* which respectively keep the data sent out from the Master and the data of the estimated state of the process. The most recent calculated data is inserted at the beginning of the lists so that it is easier to find the right data we need and delete the useless ones.

(a) ConsThread gets the consign (it is the position where the user wants the motor to arrive) from a file given by the user. In this way, the user can freely change the task. The time period T_3 for this thread to work continuously is also set by the user, e.g. 5 seconds.

(b) SenderThread gets the different consign ($c(t_{3,p})$) every time period of ConsThread. Then it calculates the control data to send out according to the following equation:

$$u(t_{1,k}) = K(\hat{x}(t_{0,q}) - c(t_{3,p})). \quad (11)$$

The most recent $\hat{x}(t_{0,q})$ can be found at the beginning of the *list_X* ; then, the data of command together with the system time is sent out to the slave through the socket. While, at the same time it is inserted into the *list_U* for the ObserverThread to use.

In order to adjust the $u(t)$ with the value of $x(t)$ which is the estimated state of the motor, the time period of this thread should be chosen much smaller than that of ConsThread, here 0.1 second is applied.

(c) ReceiverThread is a event-driving thread. As there is data arrived from the slave, it first check whether there is packet loss. As the time period for the Slave to send out the data is so small that several packets lost will not affect much. Then according to the time stamp, the most recent data is sent to the thread of ObserverThread.

(d) ObserverThread is the most important part of the program. It mainly servers as the Observer in the system model. The main task is to calculate the most possible position and speed of the motor. To work out this, it is needed to find out the command u which has been applied to the slave system and the estimated motor position at the time when the information is sent out from the slave.

As it is illustrated in Fig.4, in order to determine $\hat{y}(t_{2,k'})$, it is necessary to find in the *list_X* the closer state estimation $\hat{x}$ with regard to the date $t_{2,k'}$. And we can get the control data u in the *list_U* with the time stamp of time h_{1m} before. So, according to the equation (4), the state estimated can be obtained. As

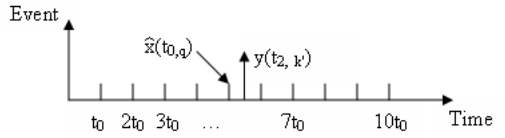

Fig. 4. Packet Sequences

we can see from the Fig.4, in order to find the state $\hat{x}(t_{0,q})$ at the time nearly to $t_{2,k'}$, the time period of this thread should be small enough. We choose here 0.02 second. As the results showed after, it is sufficient.

4.2 The structure of the Slave

The Slave does not need power computation abilities, which is designed to communicate with the Master and the Miabot. As we can see from Fig.5, this program is divided into two threads: ReceiveThread and SendThread. As we need to apply the control data the time delay of h_{1m} after the time stamp, a *list_Y* is used to contain the control data temporarily.

(a) ReceiveThread is an event-driving thread which is activated by the control data arrived from the Master. The control data is inserted into the proper position of the list *list_Y* according to its time stamp. If the time stamp is before the oldest data of the list, that

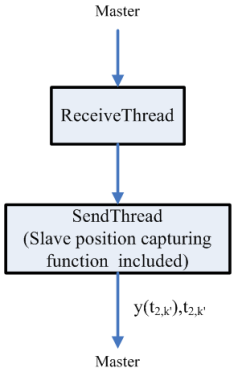

Fig. 5. Structure of the Slave

means there is disorder of the packets through Internet, then the data is discarded. If there are several packets lost, as we have a high frequency of the Master's SenderThread, it has little effects on the whole performance of the system.

(b) SendThread includes a function to get the real position of the Miabot. Considering the character of the motor, we choose a time period T_2 for this thread as 0.1 second. In every circulation, it look for the control data to be applied in *list_Y* and then send it to the Miabot by the port of Bluetooth. The state data of the Miabot is then sent to the Master.

5. RESULTS AND ANALYSIS

After identification of the Miabot, we get the model of the following form

$$\begin{cases} \dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & -25.6 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 2.56 \end{bmatrix} u(t-\delta_1(t)), \\ y(t) = \begin{bmatrix} 1 & 0 \end{bmatrix} x(t). \end{cases}$$

We have continuously tested the RTT (round-trip-time) between the two PCs in the daytime and night-time by the protocol ICMP (Internet Control Message) as respectively showed in Fig.A.2 and Fig.A.3. According to Fig.A.4, more than 95% round-time-trip(RTT) are less than 60ms. From these tests, an evaluation of the maximal time delay is obtained. So in the system, considering the characteristics of Internet, Bluetooth transmission delays and the time delay for applying the strategy combined with the sampling time, we take the value $\delta_1 = \delta_2 = 0.08sec.$,and $\mu 1 = \mu 2 = 0.02sec.$

The gains K and L have to be computed in such a way they exponentially stabilize the global Master-Slave-Observer system despite the variable delays $\delta_1(t)$ and $\delta_2(t)$. We get $\alpha = 3$ if the gain L is chosen as:

$$L = \begin{bmatrix} -0.9119 \\ -0.0726 \end{bmatrix}.$$

The gain K is chosen as:

$$K = \begin{bmatrix} -0.9125 & -0.0801 \end{bmatrix}.$$

5.1 Result obtained in Matlab/Simulink

The results in Fig.6 and Fig.7 are obtained from Matlab/Simulink according to the delay variation law. On Fig.6, the continuous model of the observer $\hat{x}$ corresponds to the blue and red curves, while the sampling instants correspond to the blue and red dots. The blue output is driven to its set value (dark-blue steps).

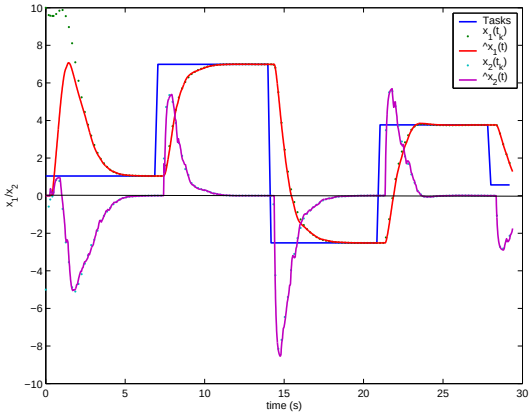

Fig. 6. Simulation results

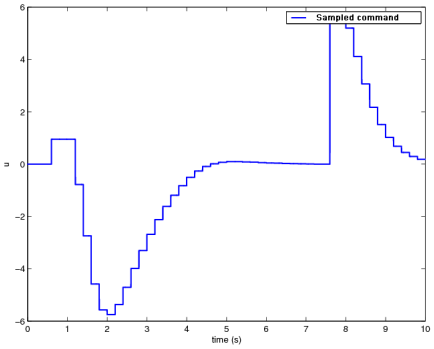

Fig. 7. The corresponding Slave control

5.2 Result of centralization experiment

In the experiment of the Master-Slave program on the same computer, 0.1 second was chosen as the maximum time delay h_{1m} which includes sampling and transmission delays. The result is shown in Fig. 8, in which the blue curve represents the consign tasks; the red and green represent motor estimated state; the dot lines correspond to the real motor state. Fig.9 represents the sampled control data send to Slave.

In this experiment, the system clock is unique, so the situation is totally in accordance with the theory and the results are similar to the simulation ones.

From the graph Fig. 8, we can see that the time instance of the consign being implemented has a little retard in accordance with h_{1m}.

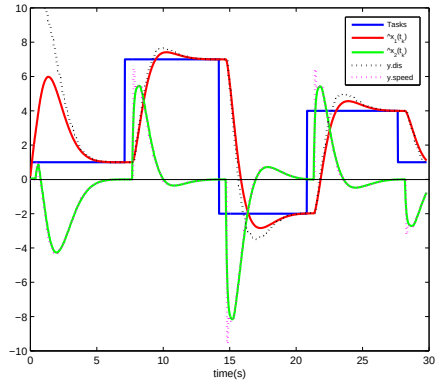

Fig. 8. Results of centralization experiment

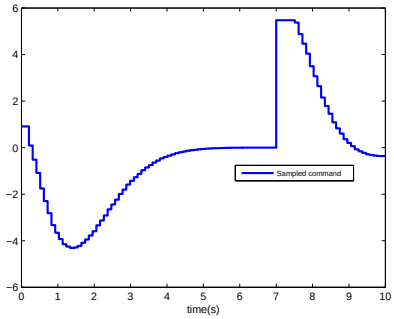

Fig. 9. The corresponding Slave control of centralization experiment

5.3 Result of remote experiment

The experiment is done on two computers separated about 40 kilometers away. We first synchronized the time of two computers by the NTP(Network Time Protocol). We choose the same time pool server, europe.pool.ntp.org. Then the Master program runs on the remote computer with an advanced computing capability, the slave program on the local one together with the Miabot. We get almost the same figure as these results of centralization experiment.

6. CONCLUSION

The experimental results confirm the theory and exponential stability is obtained. Both in the local and distance experiments confirm the theory and the Miabot follows the given instructions. To be adapted to more general situation of the Internet, we can augment grand h_{1m} while diminishing the value of α. When α is fixed as 0, we get the most grand time delay through the Internet which is tolerated by the system, h_{1m} should be no more than 1.9sec. Note that, we make the experiment after synchronization by NTP, but after running a long time, we can not guarantee the perfect same time in the two PCs. So, we consider two ways to solve the problem. One way is to find a solution for synchronizer two PCs in the long term, another way is to include the clock difference as a form of additional

time delay and to take it into account in the control. The work is on progress.

Appendix A. THE FIGURES

A.1 The figure of time delay by Bluetooth

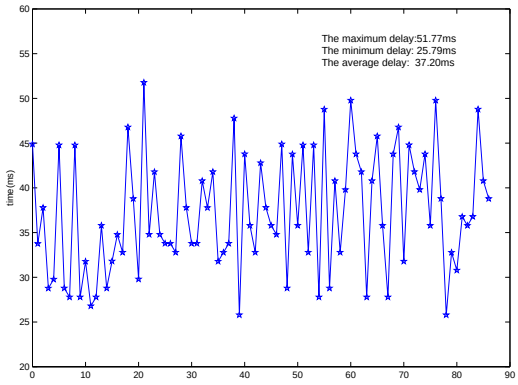

Fig. A.1. The round-trip-time(RTT) between the Miabot and the PC by Bluetooth

A.2 The figures of time delay by Internet

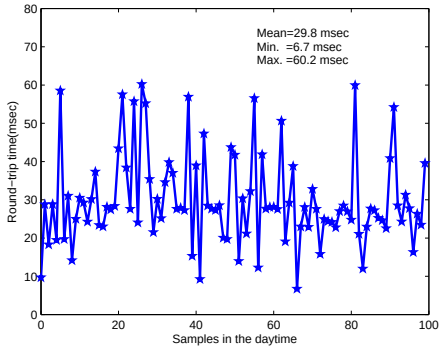

Fig. A.2. The RTT in the daytime between the two PCs by Internet (40km aways).

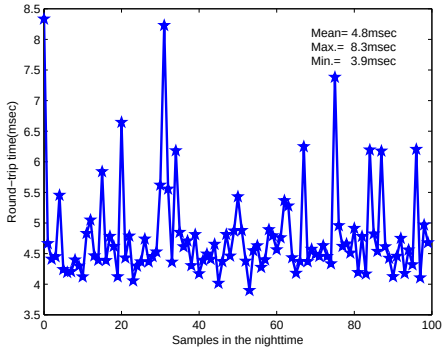

Fig. A.3. The RTT in the nighttime between the two PCs by Internet (40km aways).

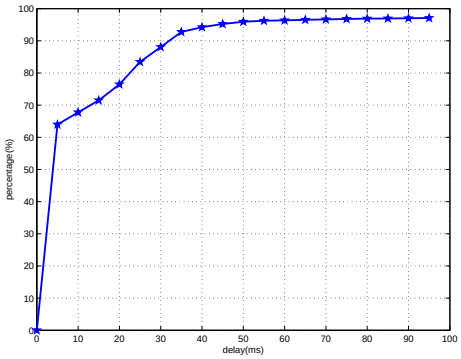

Fig. A.4. The percentage of the RTT in different time delay ranges

REFERENCES

A. Lelevé, P. Fraisse and P. Dauchez (2001). Telerobotics over IP networks: Towards a low-level real-time architecture. *IROS'01 International conference on intelligent robots and systems,Maui,Hawaii*.

A. Seuret, E. Fridman and J.P. Richard (2005). Sampled-data exponential stabilization of neutral systems with input and state delays. *Proc. of IEEE MED 2005, 13th Mediterranean Conference on Control and Automation, Cyprus*.

A. Seuret, F. Michaut, J.P. Richard and T. Divoux (2006). Networked control using gps synchronization. *Proc. of ACC06, American Control Conf., Mineapolis, USA*.

A. Seuret, M. Dambrine and J.P. Richard (2004). Robust exponential stabilization for systems with time-varying delays. *Proc. of TDS04, 5th IFAC Workshop on Time Delay Systems, Leuven, Belgium*.

D.L. Mills (1995). Improved algorithms for synchronizing computer network clocks. *IEEE/ACM Transactions On Networking* **3**(3), 245–254.

E. Fridman, A. Seuret and J.P. Richard (2004). Robust sampled-data stabilization of linear systems: An input delay approach. *Automatica* **40(8)**, 1441–1446.

E. Witrant, C. Canudas-De-Wit and D. Georges (2007). Remote stabilization via communication networks with a distributed control law. *IEEE Transactions on Automatic control*.

J. Chiasson and J.J. Loiseau (2007). *Applications of time delay systems*. Vol. 352. Springer.

J.P. Richard (2003). Time delay systems: an overview of some recent advances and open problems. *Automatica* **39**, 1667–1694.

J.P. Richard and T. Divoux (2007). *Systèmes commandés en réseau*. Hermes-Lavoisier, IC2, Systèmes Automatisés.

M. Yu, L. Wang and T. Chu (2004). An LMI approach to network control systems with data packet dropout and transmission delays. *MTNS '04 Proc. of Mathematical Theory Networks and Systems, Leuven, Belgium*.